EUROPEAN PATENT APPLICATION

(12)

(72) Inventor: Jones, Gardner Dulany, Jr.
4711 Tanglewood Drive
Raleigh, NC 27612(US)

(74) Representative: Bonneau, Gérard
Compagnie IBM France Département de
Propriété Intellectuelle
F-06610 La Gaude(FR)

(54) **Signal processor with independently arithmetic and logic unit and multiplier accumulator unit simultaneously operable.**

(57) The architecture of the signal processor operates the ALU (10) and MACU (11) through a register file (9) that serves as a general buffer pool for operands. All operand transfers take place between data memory through this register file (9) and ALU (10) and MACU (11) have equal access to all data in the file (9). Further the file (9) is the buffer for previous ALU results. In this manner, the bandwidths of all the individual units, data buses (19, 20), ALU (10) and MACU (11) can be fully utilized without conflicts. In general, the proposed configuration relies on the redundance or latency in many signal processing computations where data and results are used and reused in the overall computation and must remain in holding registers. The register file gives this capability providing these operands for use independently by both the ALU (10) and MACU (11). Without a common register file, operands would have to be reloaded as the computation continues. These redundant loads reduce the throughput for the computation.

FIG. 1
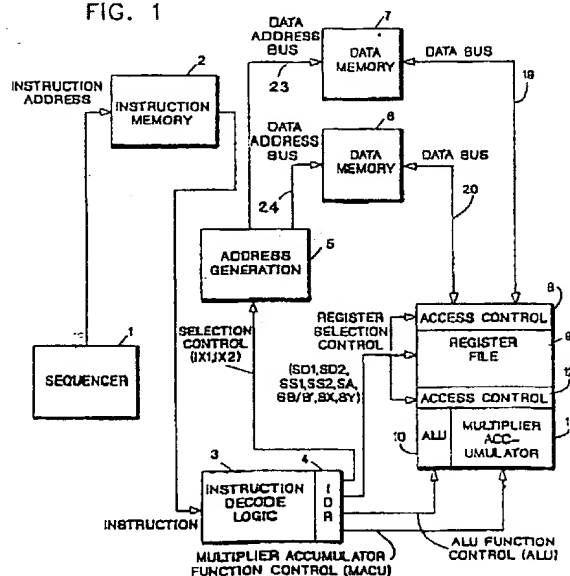
EP 0 425 410 A2

SIGNAL PROCESSOR WITH INDEPENDENTLY ARITMMETIC AND LOGIC UNIT ANO MULTIPLIER ACCU-MULATOR UNIT SIMULTANEOUSLY OPERABLE

The invention relates to data processing systems in general and particularly to a signal processor adapted for performing rapid, repetitive calculations for Fourier transformations, digital filters, compression coding, correlation, equalization modem functions, speech recognition synthesis or compression and image recognition or enhancement as well as instrumentation and filtering, thanks to independently simultaneously operable arithmetic and logic unit and multiplier accumulator unit.

A variety of commercially available, high speed signal processors currently exists. For example, U.S. Patent 4,794,517 describes a three-phased pipelined signal processor architecture. This architecture is capable of numerous high speed operations and does have a separate arithmetic and logic unit (ALU) as well as a separate multiplying function. However, the multiplier in this prior patent does not have accumulator registers or dual data bus structure that would permit independent and simultaneously concurrent operation of the ALU and multiply accumulate functions performed by the multiplier accumulator (MACU).

Similarly, VanWijk et al have described a digital signal processor with parallel processing capability in an article entitled "A Two Micrometer C Moss 8 MIPS Digital Processor with Parallel Processing Capability" appearing in the IEEE Journal of Solid State Circuits, Volume SC-21, Number 5, October 1986, page 750 et sic.

This processor does exhibit separate MACU and ALU units but concurrent operation of the ALU and MACU with high rate of memory transfer functions is not possible because of the bus and register configuration employed. When two operands must be transferred from data memory to the ALU via the X and Y buses, the MACU is prevented from having any input either from memory or from the ALU because the data buses are occupied with the operand transfers to the ALU. A simple simultaneously concurrent operation such as the adding of two operands by the ALU and multiplication of the result by a third operand in the MACU cannot be performed because no transfer paths are available to operate the MACU. Such a configuration requires an additional machine cycle to take the results from the ALU output registers, transfer them via data buses and place them in input registers of the MACU for the multiplication step. Such an architecture does operate the ALU or the MACU independently but they may not be operated simultaneously or concurrently. This architecture thus achieves no performance gain from the separation of the two functions of the ALU and MACU.

Another commercially available signal processor having a separate ALU and MACU is the Analog Devices processor described in the Analog Devices Users Manual E971-10-4/1986. This architecture operates the ALU and MACU separately but not concurrently. This architecture does have an additional result bus for transfer of previous results between the ALU and the MACU without blocking the data buses. However, only the previous result from either the MACU or the ALU can be transferred across the result bus in this manner, and it provides only a single result operand to the other unit, but not two operands. When two operands are transferred to the ALU via data buses, the output of the ALU can provide one input to the MACU. The other operand must have been loaded in a MACU input register previously either from data buses or from the ALU.

The limited utility that this configuration offers for simultaneous operation is probably the reason that the vendor provides no capability in the instructions for carrying out simultaneous operations. The ALU and the MACU are operated separately and independently, but not concurrently. Thus no performance gain is achieved by the configuration. Even if a more general routing capability were provided between the ALU and the MACU for both of the foregoing vendor architectures, they would still not achieve the maximum possible throughput that separating the processing functions themselves can offer because both of these prior ALU and the MACU devices have separate input register files. In both of these architectures, therefore, the processing units ALU and MACU do not have common access to the operands once they are transferred from data memory. To have common access to the operands, the operand must be loaded in parallel into input registers in both the ALU and the MACU or an additional machine cycle must be taken to perform the transfers.

In light of the foregoing known difficulties with the prior art signal processors and their architecture, it is an object of the present invention to provide an improved signal processor having independently and simultaneously operable ALU and MACU units.

The improved signal processor in the present invention permits simultaneous operation of the ALU and MACU by providing a register file that serves as a general buffer or pool or operands. All operand transfers take place between separate data memories through two separate buses into the register file or out of the file into the memories. The ALU and MACU have equal access to all the

data in the file at any time. The file also serves as a buffer for any previous ALU results. It is not necessary to have the output of the MACU available in the register file because the ALU performs operations for results on a single operation basis and the output from the MACU will be available for each operation of the ALU if required. In contrast, the MACU may take many operations to produce a meaningful result. The register file gives the capability of providing operands for use independently by both the ALU and the MACU, without which operands would have to be reloaded as the computation continues, thereby causing a redundant transfer load that will reduce the throughput of computation. In order to avoid reloading of operands, operations in the ALU must be possible in the form of R = A as a function of B. In this form, a third operand the result R, is generated without destroying either of the input operands. This may be contrasted with the conventional operations of the form A = A as a function of B in which one of the input operands, A, is replaced by a new result A. The significance of the improved instruction form is that it eliminates reloading an erased operand for use by a subsequent operation within a few cycles which would cause extra data bus conflicts and reduce the throughput. Independent and concurrent operation of the ALU and MACU require that instructions calling for the improved mode of operation must have separate control fields for each unit as well as control fields for the register file input and outputs.

The invention will be described with reference to a preferred embodiment which is further illustrated in the following drawings in which :

Figure 1 illustrates a high level schematic flow diagram of the main operational portions and their interconnection in constructing a signal processor having separate, independently and simultaneously operable ALU and MACU units.

Figure 2 illustrates that portion of an improved signal processor in greater detail showing the interconnection of the ALU and MACU to the register file and the separate address generation and data memory paths as utilized in the preferred embodiment.

Figure 3A illustrates a typical instruction decode register format for a load and compute operation.

Figure 3B shows a typical instruction decode register format for a store and compute operation.

Figure 4 shows the chief operational elements of an improved signal processor incorporating the register file and independent ALU and MACU units together with the control interconnections from the instruction decode logic and main system clocking signals.

Figure 5A illustrates schematically the timing of operation for the architecture of the instruction processor in the preferred embodiment in a non-pipelined configuration.

Figure 5B illustrates schematically the clocking and sequencing for a three-phased pipelined mode of operation of the same architecture.

Figure 6 illustrates input selection for the registers in the register file together with the selection and decode logic.

The present invention is directed to the architecture of separately and simultaneously operable ALU and MACU units together with a register file serving as a buffer pool for all data transfers to and from memory and the operative ALU and MACU units. Therefore, the degree of detail for such elements as a sequencer for instruction addresses for instruction fetching, the details of instruction decode logic or address generation, etc. are not given herein as such details are readily available in the prior art and are well understood by those of skill in the art.

Turning to Figure 1, the overall schematic layout of the operative units constituting a signal processor having independently and simultaneously operable ALU and MACU is shown.

The sequencer 1 steps the instruction memory 2 through a list of preloaded instructions to provide them individually when required or "fetched" to the instruction decode logic 3. Incoming instructions are decoded and the results loaded into an instruction decode register, IDR 4. Various output control lines exit from the IDR 4 to set the function of the ALU and the MACU and to control I/O access of data and operands to and from the register file 9 via the access control logic 8 and 12. IDR 4 controls the ALU 10 and MACU 11 as well as register file 9 and the access control 8 and 12, respectively. A selection control line pair goes to the address generation 5 which provides data address inputs on data address buses 23 and 24 into the data memories 6 and 7, respectively. The data memories 6 and 7 provide their output data on buses 19 and 20 as shown on Figure 1 to the access control 8 to gain entrance to or exit from register file 9. The construction of all of these units in Figure 1 is a matter well known in the prior art and details of the construction of the elements themselves, with the exception of some of the details for the register selection control means 8, 12, the IDR 4 and the address generation 5, is therefore not given.

Turning to Figure 2, a more detailed representation of the major components of the group comprising the register file 9 and the separate ALU and MACU 10 and 22, respectively, together with the dual data memory and address bus structure as well as the dual address generation structure for the data memories are depicted with the input

control lines and clocking omitted for clarity. It may be observed that the data memories 6 and 7 are individually driven by their address generators 5A and 5B and provide independent outputs on the data buses 19 and 20 to the register file input control means for loading 8A or to the control 8B for reading from the register file. Additionally, outputs from the register file may be selected as an operand via the output control 12 or the output of the ALU may be selected through another portion of the control 12 for input to the register file 9. Output control 8C controls the selection of either the output selection B or B'. Output B' can be the output of the multiplier 13 or of either of the multiplier accumulator registers 15A and 15B as controlled by the output from the IDR 4 through the selector 8C which acts as the access control means for this portion of the operand selection. It may be seen that the output from the register file 9 may be selected for input to the ALU 10 and that either another output from register file 9 or the result of the multiplier 13 or some addition or subtraction result which appears in the accumulator registers 15A or 15B may be selected as the operand B or B' for the other input to the ALU 10. The operand selection for B or B' is via operand register file access control 8C under the control of IDR 4 as will appear in greater detail later.

It may be noted that the multiplier 13 provides an output to a 32-bit adder 14 whose input is also received from a 32-bit selector 16 under control of the IDR for recirculating the content of the accumulation register 15B for further mathematical operations in the adder 14, again under the control of the IDR. Separate multiplication and accumulation results may therefore be stored separately in the registers 15A and 15B depending upon selection signals from the IDR (not shown).

Also the address generation logic for the two data memories 6 and 7 are separate and independent. The address generator 5B generates an address for data memory 7 and has separate incrementation and displacement registers 18B and 17B, respectively, which may be utilized in generating the addresses for the data memory 7. A similar structure exists separately for the address generator 5A together with its separate displacement register 17A and incrementation register 18A. It provides addresses to the data memory 6. The independent data memories with their independent address generators provide independently their results or output on the data buses 19 and 20, respectively. Thus, on any clock cycle, two operands from data memories 6 and 7 may be simultaneously presented to the register file 9 or two outputs from register file 9 may be selected for input to the data memory or recirculation back into the register file. At the same time two separate

outputs X and Y may be selected for input to the MACU, two outputs may be selected for input to the ALU and an input from the ALU may be selected for simultaneous input to the register file.

Figure 3A illustrates a typical instruction decode register (IDR) 4 content for an eight-register file 9 for a load and compute operation. The assignments of the control fields in the IDR 4 and the functions that may be selected are shown in Figure 3A. It may be observed that the ALU control function includes some functions of the form $C = A$ function of B in which the result of the ALU output is stored in a new register C without destroying either of the original input operands A or B.

The field for address selection and control can use either the incrementation register or the incrementation register plus a fixed displacement that may be selected between two different values or two different incrementations depending upon the selection control code.

The MACU control field contains eight possible specified operations of the form as shown.

Figure 3B shows the typical instruction decode register content for an eight register file in a store and compute operation in which the output store selection controls are provided to the file register access selectors 8B shown in Figure 2.

Figure 4 illustrates in somewhat greater detail the clocking and controls for operating the register file access control elements (8A, 8B, 8C 12) under control of the IDR 4 and also shows the IDR 4 controlling the address generators 5A and 5B, the ALU 10 and the MACU 11. The clock 21 provides its clocking signals simultaneously as shown to all of the operative units including the register file 9, the address generators 5A and 5B, the MACU unit 11 and, though not shown, would also be supplied to set the accumulators 15A, 15B at the output of the MACU and to the instruction decode logic 3 which sets its output in the IDR 4.

The architectural layout of the elements as shown in Figures 1, 2 and 4 may be operated either in a pipelined or non-pipelined mode of operation.

Figure 5A illustrates the sequencing with the regularly occurring pulses from clock 21 for operation of the architecture in a non-pipelined operation including steps of "fetch an instruction", "decode an instruction", "transfer any data" necessary for executing the instruction and, finally, "computing or executing" the instruction. The timing of each operation is seen to be at the boundary which is the clock pulse ending a given clock period and beginning the next clock period.

Figure 5B illustrates the timing and sequencing for a three-phase pipelined mode of operation in which the data transfer and computation actions are taken in the same clock cycle for an instruction

that has been decoded in the previous clock cycle and which was fetched on the clock cycle preceding that.

Register input and output selectors for the file register access control can be made of standard Texas Instrument part number 54ALS151 utilizing 16 eight-way selectors. Figure 6 illustrates schematically the set of controls for what will be assumed to be an 8 x 16 array of file registers. The individual control fields of the IDR 4 are decoded in the decoders 22 A-N to operate through the gating logic to control setting of registers 1-8 in the register file 9 as schematically shown.

The described architecture and data flow in which the ALU and the MACU are separate and may be independently and simultaneously operated in the same clock cycle permits multiplication and accumulation of sums to be done independently of any operations going on in the ALU and vice versa. By permitting independent control of the ALU during such operations, a factor of at least two increase in throughput may be realized over that provided by many commonly used signal processing algorithms and machines. The data flow has several advantages in terms of implementation. First the ALU and register file are located in a single precision portion of the data flow while interfaced to the data buses is of the same word length. Only the adder and accumulator registers need to have double precision necessary for product accumulation and the output of the accumulators may be taken into the data flow as single precision operands (16 bit) instead of 32 bit double precision operands. The data flow architecture separates the less complex and high precision signal processing tasks such as product accumulation from the more complex control and microprocessor tasks performed by the ALU. The throughput gain for the data flow is achieved by operating the ALU independently while forming the sum of products separately in the MACU unit. This independent operation results in increased throughput for general operations and most obviously of the type exemplified by Equation A:

A(i) [B(j) + C(k)]

When sums of the type shown in Equation A are computed, the intermediate sum or difference is formed by the ALU and the sum of products is formed by the adder accumulator in the MACU.

Equations of the form A are useful in computing output of filters having symmetrical impulse responses, which include the linear phase filters that have even symmetry and Hilbert filters that have odd symmetry. When computing filters of this type, the sum of equation in the form A produces an effective throughput of two filter taps per processor cycle.

Adding a second data memory and data bus to the architecture does not give a meaningful increase in throughput unless the ALU and MACU functions are made separately operable simultaneously.

Equations of the type or form shown in Equation A may use three operands per cycle to perform computations and are transferred by sharing the data buses through the register file as depicted herein. Since the two data memories are physically separated the transfers of data among them and the operation units requires two address pointers per bus, care must be taken in organizing the data arrays in the two memories. Typically, coefficients can be placed in one memory and data samples in the other. However, in the typical example of auto-correlation, two samples must be accessed per clock cycle from the same data array. In this case, the data is written in both memories. The organization of data may also be determined, by whether the ALU overwrites one of its own operands when executing an operation. The organization of data in a memory is more easily done if the ALU can perform the operation C = A + B as opposed to A = A + B. The latter function may require a reload of one of the operands, thereby increasing the overall transfer load and decreasing the throughput. The capabilities of processors constructed according to this architectural design are numerous as pointed out in the beginning here under the heading "Field of the Invention" and permit operation of the ALU and MACU independently and simultaneously to utilize the maximum transfer bandwidth of the dual data memory buses.

Numerous changes in implementation without departing from the spirit and scope of utilizing dual data bus memory architecture with separately and simultaneously operated MACU and ALU units in a signal processor will easily be suggested, wherefore what is desired to be protected by letters patent and what is claimed is set forth by way of example and not by way of limitation.

Claims

1. A signal processor having an arithmetic and logic unit ALU (10) and a multiplier accumulator unit MACU (11) which operate in a independent and simultaneous way, and comprising a system clock (21), a random access register file, register file access control means (8,12), two independently operable random access data memories (6,7), and two independently operable data buses (19,20) connected with said register file access control means and with said two random access data memories; said processor being characterized in that :

Said register file (9) is connected to said data

buses, to said MACU (11), and to said ALU (10)- through said register file access means so that said data memories, said MACU and said ALU may simultaneously gain access to any register or registers in said register file, and

all data transfers between said data memories and said ALU and said MACU is made by loading the data to be transferred into said register file for access thereto by the intended recipient of said data.

2. The processor as described in claim 1, further characterized in that said system clock (21) supplies a single clock signal simultaneously to all of the clocked elements in said processor.

3. The processor as described in claim 1, further characterized in that at least two independent accumula tor registers (15A, 15B) connected in parallel at the output of said MACU (11) receive the output results from the operation of said MACU.

4. The processor as described in claim 1, 2 or 3 further comprising two independent data memory addressing control means (5A, 5B) for generating two independent data memory addresses.

5. The processor as described in claim 5, further characterized in that each of said data memory addressing control means (5A, 5B) operates with at least two index registers (18A, 18B) and at least two displacement registers (17A, 17B).

6. The processor as described in any one of claims 1 to 5, further comprising an instruction memory (2) , an instruction memory address control sequencing means (1) connected to said instruction memory, and an instruction decode logic means (3) connected to said instruction memory to receive and decode instructions therefrom and further comprising an instruction decode output register (IDR) (4) connected to said memory addressing control means, to said register file access control means, and to said ALU (10) and to said MACU (11) for controlling register selection of said register file (9) for input or output of data and for controlling the functions of said ALU and of said MACU, and

wherein said IDR (4) has register segments for holding decoded instructions comprising at least two separate memory address control field segments, two separate register file data input selection control field segments, four separate register file output control field segments and at least one separate ALU function control field segment and at least one separate MACU control field segment and at least a separate functional control field segment for selecting the non-destructive retention of ALU result operands.

7. The processor as described in claim 6, wherein said IDR control field for said ALU specifies a selection of arithmetic controls of the form A = B as a function of C where the arithmetic function specified by function includes at least addition and

subtraction.

8. The processor as described in claim 7, wherein said IDR control field for said MACU specifies a selection of arithmetic controls of the form Accum1 = Accum1 + a product P.
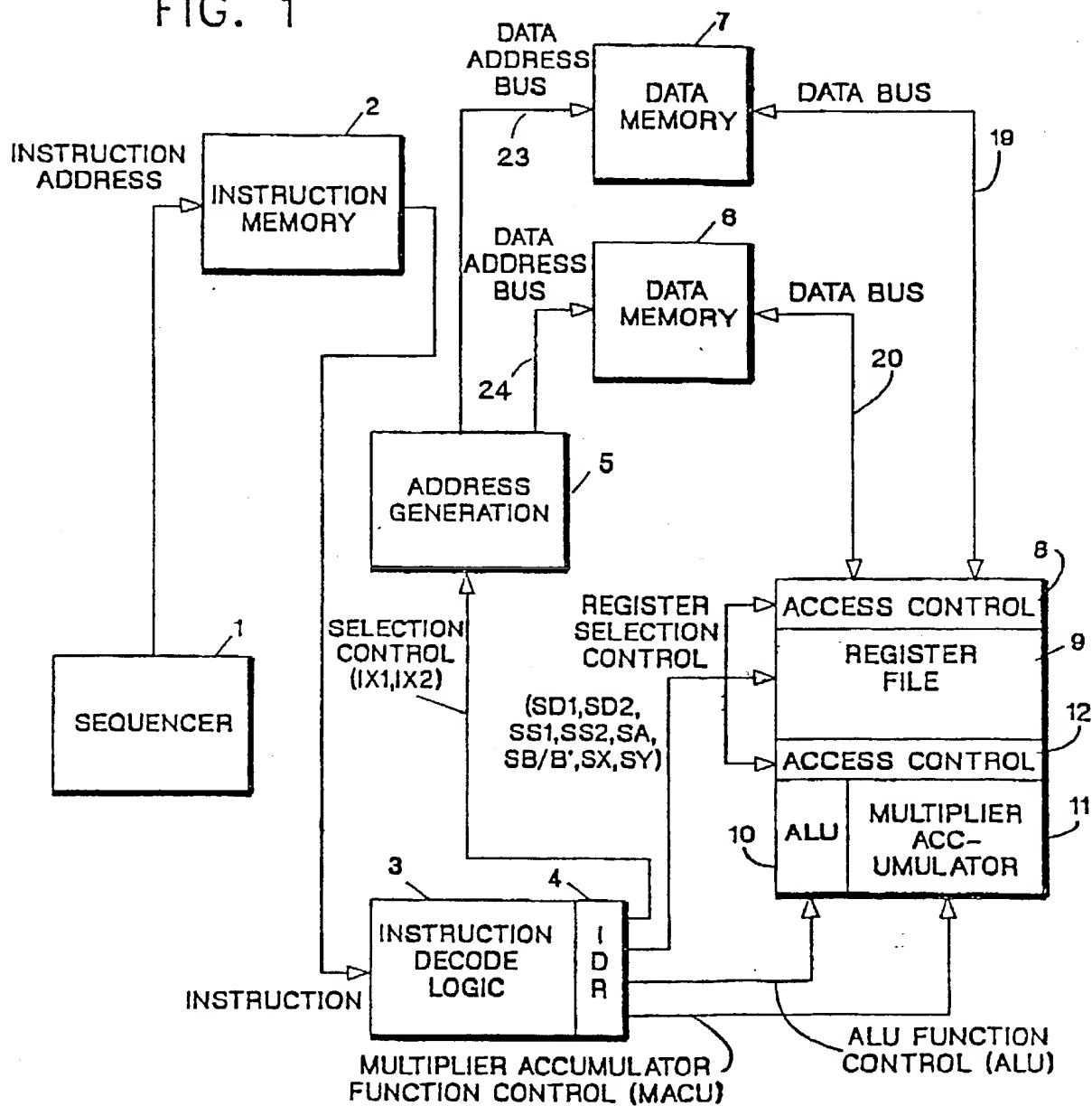
FIG. 1

FIG. 2

5B	5A	TO 8A		12	12	8C	12	12	12	10	11

IX1	IX2	SD1	SD2	SA	SB/B'	SC	SX	SY	ALU	MAC
0 1	0 1	0 1 2	0 1 2	0 1 2	0 1 2 3	0 1 2	0 1 2	0 1 2	0 1 2	0 1 2

LOAD
REGISTER
SELECTION

ALU
OUTPUT
SELECTION

MAC
CONTROL

AC1=AC1+P
AC1=AC1−P
AC1=   P
AC2=AC2+P
AC2=AC2−P
AC2=   P
AC1=AC2=0
AC1=AC2=P

ADDRESS
SELECTION
AND
CONTROL

ALU
INPUT
SELECTION

MULTIPLIER
INPUT
SELECTION

ADD1 = IR1
ADD1 = IR1, IR1=IR1+Dr1
ADD1 = IR2
ADD1 = IR2, IR2=IR2+Dr2

ALU
FUNCTION
CONTROL
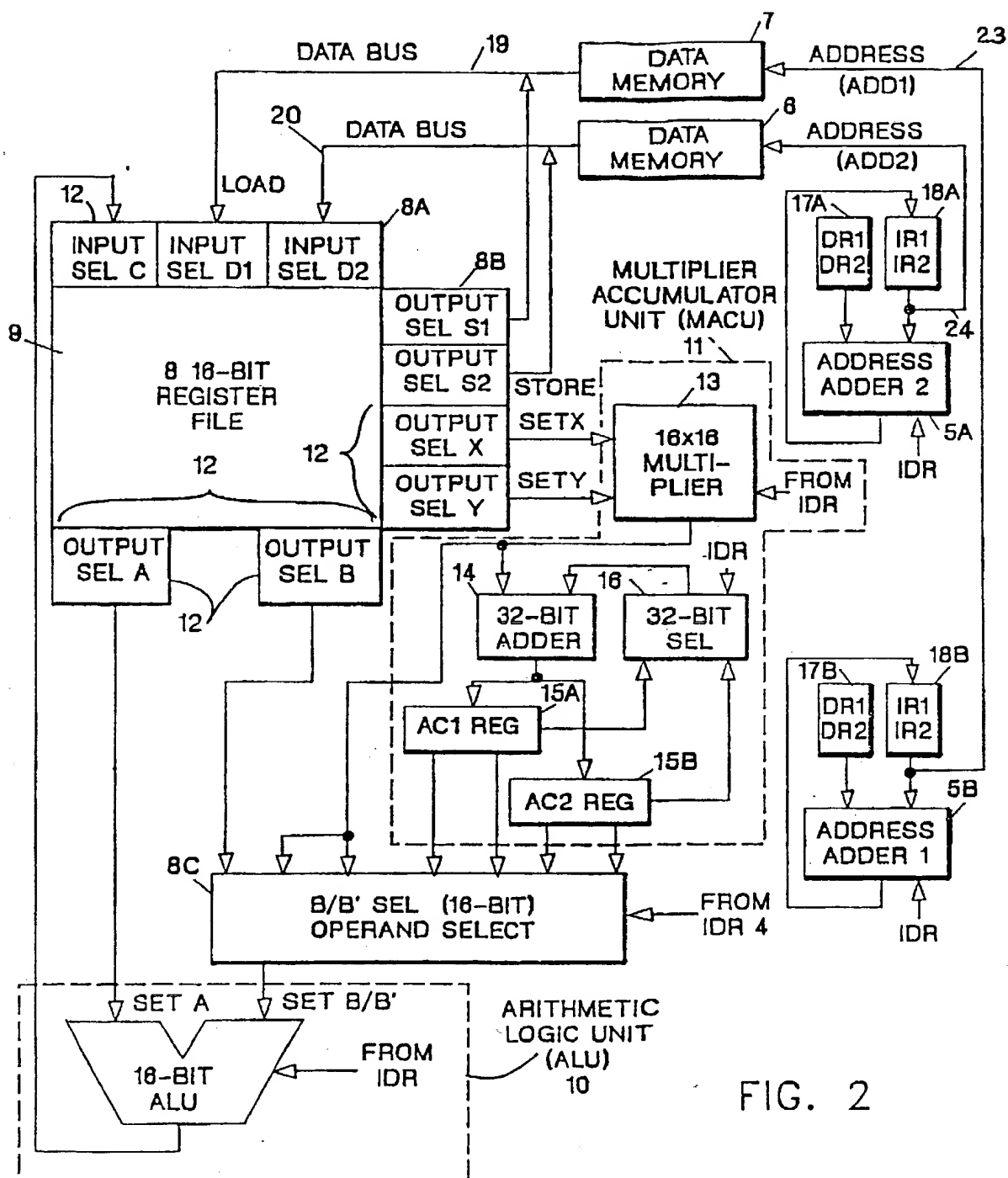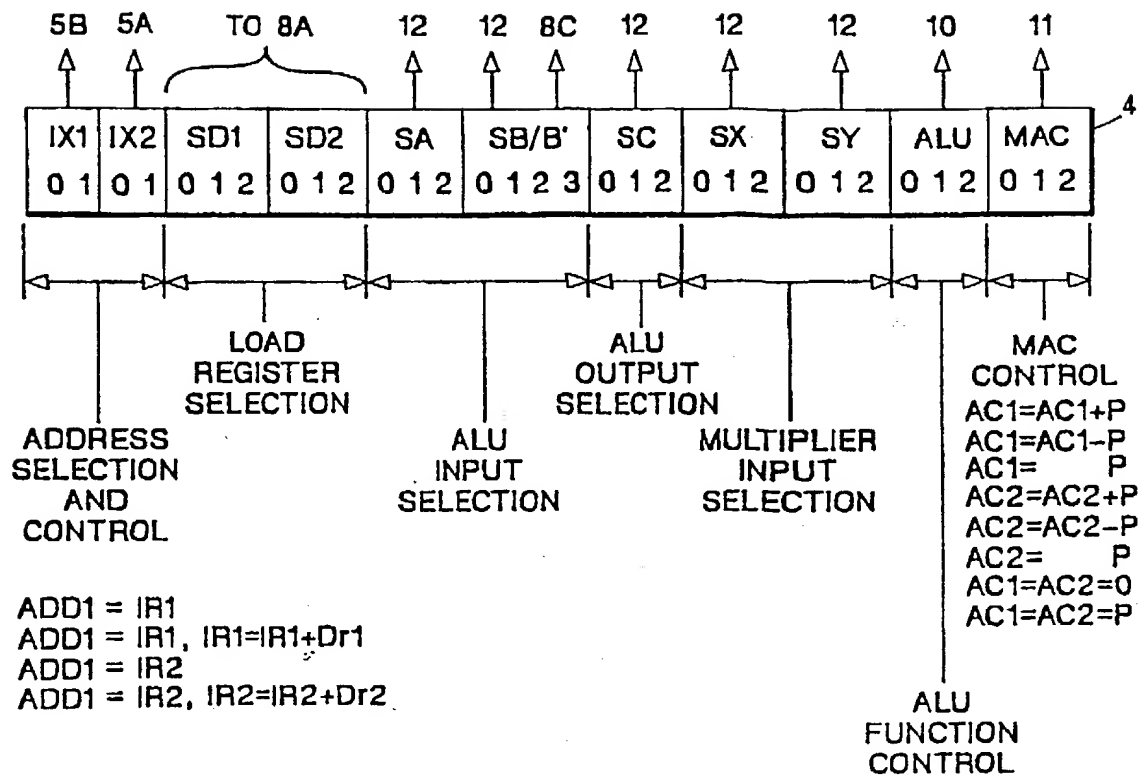
A = A + B
A = A + B + CARRY
A = A − B
A = A − B + CARRY
C = A + B
C = A + B + CARRY
C = A − B
C = A − B + CARRY

FIG. 3A

TO 8B

TO 12

IX1	IX2	SS1	SS2	SA	SB/B'	SC	SX	SY	ALU	MAC
0 1	0 1	0 1 2	0 1 2	0 1 2	0 1 2 3	0 1 2	0 1 2	0 1 2	0 1 2	0 1 2

STORE
REGISTER
SELECTION

FIG. 3B

FIG. 4

FIG. 5A

FETCH	DECODE	TRANSFER	COMPUTE
LOAD INSTRUCTION FROM INSTRUCTION MEMORY → SET INSTRUCTION REGISTER	DECODE INSTRUCTION → SET INSTRUCTION DECODE (IDR) REGISTER → GENERATE DATA MEMORY ADDRESSES	ACCESS·(TO/FROM) DATA MEMORIES → SET ARITHMETIC REGISTERS (ALU & MACU)	OPERATE ALU (COMPUTE) OPERATE MAC (COMPUTE) → SET ARITHMETIC REGISTERS

| T CLOCK PERIOD | T CLOCK PERIOD | T CLOCK PERIOD | T CLOCK PERIOD |

FIG. 5B

INSTRUCTION N

FETCH	DECODE	TRANSFER COMPUTE

INSTRUCTION N+1

FETCH	DECODE	TRANSFER COMPUTE

INSTRUCTION N+2

FETCH	DECODE	TRANSFER COMPUTE

INSTRUCTION N+3

FETCH	DECODE	TRANSFER COMPUTE

FIG. 6